クレジット:
Mathematics and Informatics Center 計算機実験I 2020 藤堂眞治

計算機実験 I (第 2 回)

藤堂眞治

2020/05/07

数値誤差の原因

- 打ち切り誤差: テイラー展開による近似を有限項で打ち切ることによる誤差 (例: 数値微分)
- 丸め誤差: 無理数や 10 進数を有限のビットの 2 進数で表現することによる誤差 (例: 0.1 が 0.10000000000000006 になる)
- 桁落ち: 非常に近い数の引き算により生じる
- 情報落ち: 非常に大きな数に小さな数を足し込む場合に生じる (例: 数値積分や常微分方程式の初期値問題で刻み幅を小さくしすぎると生じる)
- オーバーフロー (桁あふれ): 表現できる最大値を超えてしまう

桁落ち

- 2 次方程式 $ax^2 + bx + c = 0$ の解の公式

$$x_{\pm} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

  $b^2 \gg |ac|$ の時、桁落ちが生じる

- 例) $2.718282x^2 - 684.4566x + 0.3161592 = 0$ の解を 7 桁の精度で計算してみる (伊理・藤野 1985)

$$\sqrt{D} = \sqrt{(684.4566)^2 - 4 \times 2.718282 \times 0.3161592} = 684.4541$$

$$x_+ = \frac{684.4566 + 684.4541}{2 \times 2.718282} = \frac{1368.911}{5.436564} = 251.7970$$

$$x_- = \frac{684.4566 - 684.4541}{2 \times 2.718282} = \frac{0.0025}{5.436564} = 0.0004\underline{598493}$$

MATLAB での計算

- MATLAB はシンボリック変数を含む式で小数を使うと、中で分数に直して (厳密に) 計算する

```
>> syms x
>> solve(2.718282*x^2-684.4566*x+0.3161592,x)
ans =
 2568765407259397l2/2040342300381321 -
      659850729835795709785858342301921O5^(1/2)/2040342300381321
 659850729835795709785858342301921O5^(1/2)/2040342300381321 +
      2568765407259397l2/2040342300381321
```

- 10 桁の精度で解を評価してみる

```
>> vpa(solve(2.718282*x^2-684.4566*x+0.3161592,x),10)
ans =
 0.000461913553
    251.7970337
```

桁落ちを防ぐ方法

- b の符号に応じて、一方を求める (この例では x_+)
- 他方は解と係数の関係を使って求める

$$x_- = \frac{c/a}{x_+} = \frac{0.3161592/2.718282}{251.7970} = 0.0004619138$$

- 回避できない例: 重解に近い場合
  $2.718282x^2 - 1.854089x + 0.3161592 = 0$

$$\begin{aligned}\sqrt{D} &= \sqrt{(1.854089)^2 - 4 \times 2.718282 \times 0.3161592} \\ &= 0.00264575 \\ x_\pm &= 1.854089 \pm 0.00264575 = 1.856737, 1.851445\end{aligned}$$

数値微分 (差分)

- 関数のテイラー展開

$$f(x+h) = f(x) + hf'(x) + h^2 f''(x)/2 + h^3 f'''(x)/6 + \cdots$$

- 数値微分の最低次近似 (前進差分)

$$f_1(x,h) \equiv \frac{f(x+h) - f(x)}{h} = f'(x) + hf''(x)/2 + O(h^2)$$

- より高次の近似 (中心差分)

$$f_2(x,h) \equiv \frac{f(x+h) - f(x-h)}{2h} = f'(x) + h^2 f'''(x)/6 + O(h^3)$$

- 刻み h を小さくすると打ち切り誤差は減少するが、小さすぎると今度は桁落ちが大きくなる

差分の一般式

- 関数のテイラー展開: $f(x+h) = \sum_k \frac{h^k}{k!} f^{(k)}(x)$
- $f^{(m)}(x)$ を n 個の $f(x+h_j)$ の線形結合で表す $(n \geq m+1)$

$$f^{(m)}(x) \approx \sum_j a_j f(x+h_j) = \sum_j a_j \sum_k \frac{h_j^k}{k!} f^{(k)}(x)$$

$$= \sum_k C_k f^{(k)}(x) \qquad (C_k \equiv \sum_j a_j \frac{h_j^k}{k!})$$

- $C_k = \delta_{k,m}$ $(k = 0 \cdots n-1)$ となるように $a_0 \cdots a_{n-1}$ を決める
- 行列 $G_{kj} = \frac{h_j^k}{k!}$ と列ベクトル a_j と $b_k = \delta_{k,m}$ を導入すると、条件式は $Ga = b$ と書ける (連立一次方程式)

刻み幅を変えた計算

- 刻み幅を変えて何度か計算を行い、収束の様子をみる
- グラフ化して目で見てみる
- 理論式と比較
  - 計算式の正しさの確認
  - 近似の改良 (収束の加速・補外)
- 桁落ち・情報落ちの影響の有無

ニュートン法

- 反復法により方程式 $f(x) = 0$ の解を求める
- 真の解を x_0、現在の解の候補を $x_n = x_0 + \epsilon$ とすると

$$0 = f(x_0) = f(x_0 + \epsilon - \epsilon) = f(x_n) - f'(x_n)\epsilon + O(\epsilon^2)$$

- 次の解の候補 (反復法、逐次近似法)

$$\epsilon \approx \frac{f(x_n)}{f'(x_n)} \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- 複素変数の複素関数や多変数の場合にも自然に拡張可

ニュートン法の収束

- x_n が x_0 に十分近い時

$$f(x_n) \approx f'(x_0)(x_n - x_0) + f''(x_0)\frac{(x_n - x_0)^2}{2}$$
$$f'(x_n) \approx f'(x_0) + f''(x_0)(x_n - x_0)$$

- ニュートン法で一回反復すると

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \approx x_n - (1 - \frac{f''(x_0)}{f'(x_0)}\frac{(x_n - x_0)}{2})(x_n - x_0)$$
$$(x_{n+1} - x_0) \approx \frac{f''(x_0)}{2f'(x_0)}(x_n - x_0)^2$$

- 一回の反復で誤差が 2 乗で減る (正しい桁数が倍に増える) ⇒ 二次収束

多次元の場合

- $f(x) = 0$: d 次元 (非線形) 連立方程式
- x は d 次元のベクトル: $x = {}^t(x_1, x_2, \cdots, x_d)$
- $f(x)$ も d 次元のベクトル: $f(x) = {}^t(f_1(x), f_2(x), \cdots, f_d(x))$
- 真の解のまわりでの展開 $(x_n = x_0 + \epsilon)$

$$0 = f(x_0) = f(x_0 + \epsilon - \epsilon) = f(x_n) - \frac{\partial f(x_n)}{\partial x} \cdot \epsilon + O(|\epsilon|^2)$$

- ヤコビ行列 $(d \times d)$: $\left[\frac{\partial f(x_n)}{\partial x}\right]_{ij} = \frac{\partial f_i(x_n)}{\partial x_j}$
- 次の解の候補: $x_{n+1} = x_n - \left[\frac{\partial f(x_n)}{\partial x}\right]^{-1} f(x_n)$

ニュートン法による最適化

- x は d 次元のベクトル: $x = {}^t(x_1, x_2, \cdots, x_d)$, $g(x)$ はスカラー
- 勾配ベクトル: $[\nabla g(x)]_i = \dfrac{\partial g(x)}{\partial x_i}$
- 極小値 (最小値) となる条件: $\nabla g(x) = 0$
- ニュートン法で $f(x)$ を $\nabla g(x)$ で置き換えればよい
- 次の解の候補: $x_{n+1} = x_n - H^{-1}(x_n)\nabla g(x_n)$
- ヘッセ行列 (Hessian): $H_{ij}(x_n) = \dfrac{\partial^2 g}{\partial x_i \partial x_j}(x_n)$

準ニュートン法

- ニュートン法では、ヘッセ行列の計算・保存が必要
- 準ニュートン法: それまでの反復で計算した勾配ベクトルから、ヘッセ行列を近似 (B_n)
- BFGS 法 (Broyden-Fletcher-Goldfarb-Shanno)

$$B_{n+1} = B_n + \frac{y_n y_n^T}{y_n^T s_n} - \frac{B_n s_n (B_n s_n)^T}{s_n^T B_n s_n}$$

- $s_n = x_{n+1} - x_n$、$y_n = \nabla g(x_{n+1}) - \nabla g(x_n)$
- 直接 B_n の逆行列 C_n を更新することも可能

$$C_{n+1} = B_{n+1}^{-1} = C_n + \left(1 + \frac{y_n^T C_n y_n}{y_n^T s_n}\right) \frac{s_n s_n^T}{y_n^T s_n} - \frac{C_n y_n s_n^T + s_n y_n^T C_n^T}{y_n^T s_n}$$

- 他にも、SR1 法、BHHH 法、記憶制限 BFGS 法

計算をいつやめるか？

- 残差による判定
$$|f(x)| < \delta$$
- 誤差による判定
$$|x_{n+1} - x_n| < \epsilon$$
- 解 $x = x_0$ が m 重解の場合、$x = x_0$ のまわりで展開すると
$$f(x) \simeq \alpha(x - x_0)^m$$
残差が δ 程度になったときの誤差は、$\delta^{1/m}$ 程度
逆に $|x - x_0|$ が $\delta^{1/m}$ 以下になると、$f(x)$ の値がそれ以上変化しない $\Rightarrow$ m 重解の精度は計算精度の $1/m$ 桁程度しかない
- 残差による判定と誤差による判定を併用するのがよい

反復計算

- while による反復 (ハンドブック 2.2.3 節)

```
double residual = 1;     /* 残差: 適当に大きな値 */
double error = 1;        /* 誤差: 適当に大きな値 */
double delta = 1.0e-12; /* 欲しい精度 */
while (residual > delta && error > delta) {
  /* ニュートン法の漸化式 */
  /* residual と error を計算 */
}
```

残差と誤差のどちらかが欲しい精度に達したら計算を終了

- break を使う例 (ハンドブック 2.2.4 節)

```
for (;;) {
  /* ニュートン法の漸化式 */
  /* residual と error を計算 */
  if (residual < delta || error < delta) break;
}
```

初期段階における収束の改善

- Newton 法は初期値によっては収束しない
- 発散や振動を抑える方法として「減速」が有効な場合も
- 減速
  - 反復式を少し修正する

$$x_{n+1} = x_n - \mu_n \frac{f(x_n)}{f'(x_n)}$$

  - まずは $\mu_n = 1$ として計算
  - $|f(x_{n+1})| < |f(x_n)|$ が成り立たないようであれば、μ_n を半分にして再計算
  - μ_n が十分に小さくなれば、$|f(x)|$ は必ず減少する

二分法

- 反復法により一次元の方程式 $f(x) = 0$ の解を求める
- 導関数を使わず関数値のみを利用 (c.f. ニュートン法)
- 初期条件として、$f(a) \times f(b) < 0$ を満たす 2 点の組 $(a < b)$ で解をはさみ込み、領域を狭めていく
- a と b の中点 $x = (a + b)/2$ を考える
  - $|f(x)|$ が十分小さい場合: x が解
  - $f(a) \times f(x) < 0$ の場合: $[a, x]$ を新しい領域にとる
  - $f(x) \times f(b) < 0$ の場合: $[x, b]$ を新しい領域にとる
- 領域 $[a, b]$ の幅が十分小さくなったら終了
- 反復のたびに領域の幅は半分になる
- 全ての解を得られる保証はない
- 二分法の例: bisection.c

囲い込み法 (一次元の最適化)

- 関数 $f(x)$ の極小 (あるいは極大) 点を求める
- $f(a) > f(b) < f(c)$ を満たす 3 点の組 $a < b < c$ の領域を狭めていく
- $[a, b]$、$[b, c]$ の広い方 (例えば後者) を b から見て、黄金比 $[1 : (1 + \sqrt{5})/2 \approx 0.382 : 0.618]$ に内分する点を x とする
  - $f(b) > f(x)$ の場合: $[b, c]$ を新しい領域にとる
  - $f(b) < f(x)$ の場合: $[a, x]$ を新しい領域にとる
- もともとの b が $[a, c]$ を $0.382 : 0.618$ に内分する点だった場合、新しい領域の幅は、どちらの場合も 0.618
- 最初の比率が黄金比からずれていたとしても、黄金比に収束
- 黄金分割法 (golden section) とも呼ばれる

最初の囲い込み

- 1 点を選び、適当な Δx を取る
- 左右に Δx 動かしてみて、関数値が小さくなる方へ動く
- どちらに進んでも関数値が大きくなる場合には、囲い込み完了
- 小さくなった場合、その方向へ再び増えるまで Δx を倍々に増やしながら進む
- 最後の 3 点で極小値を囲い込むことができる
- 囲い込み法のプログラムの例: golden_section.c

準備: 微分方程式の書き換え

- 2 階の常微分方程式の一般形

$$\frac{d^2y}{dx^2} + p(x)\frac{dy}{dx} + q(x)y = r(x)$$

- $y_1 \equiv y$, $y_2 \equiv \frac{dy}{dx}$ とおくと

$$\begin{cases} \frac{dy_1}{dx} &= y_2 \\ \frac{dy_2}{dx} &= r(x) - p(x)y_2 - q(x)y_1 \end{cases}$$

- さらに $\boldsymbol{y} \equiv (y_1, y_2)$, $\boldsymbol{f}(x, \boldsymbol{y}) \equiv (y_2, r(x) - p(x)y_2 - q(x)y_1)$

$$\frac{d\boldsymbol{y}}{dx} = \boldsymbol{f}(x, \boldsymbol{y})$$

- n 階常微分方程式 $\Rightarrow$ n 次元の 1 階常微分方程式

初期値問題と境界値問題

- 初期値問題
  - 微分方程式において、ある 1 点に関する全ての境界条件 (初期値) が与えられているもの
  - 質点の運動など (時系列の問題)
- 境界値問題
  - 複数の点に関する境界条件が与えられているもの
  - 物体のゆがみの計算や静電場の計算など (空間的に解く問題)
- 初期値問題は初期値から逐次的に解くことが可能
- 境界値問題は初期値問題に比べて計算法が複雑

初期値問題の解法 (Euler 法)

- h を微小量として微分を差分で近似する (前進差分)

$$\frac{dy}{dt} \approx \frac{y(t+h) - y(t)}{h} = f(t, y)$$

- $t = 0$ における $y(t)$ の初期値を y_0、$t_n \equiv nh$、y_n を $y(t_n)$ の近似値とおくと、

$$y_{n+1} - y_n = hf(t_n, y_n)$$

- Euler 法
  - y_0 からはじめて、$y_1, y_2, \cdots$ を順次求めていく

Euler 法の精度

- 微分方程式の両辺を t_n から t_{n+1} まで積分 (積分方程式)

$$y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+1}} f(t, y(t))dt = h\int_0^1 f(t_n + h\tau, y(t_n + h\tau))d\tau$$

- Euler 法は、被積分関数を定数で近似することに対応

$$f(t_n + h\tau, y(t_n + h\tau)) = f(t_n, y(t_n)) + O(h)$$

- $t = 0$ からある t_f まで積分すると、反復回数 $N = t_f/h$
- $t = t_f$ における誤差 $\sim N \times h \times O(h) = O(h)$

Euler 法の改良

- 積分方程式の被積分関数をもう 1 次高次まで展開

$$f(t_n+h\tau, y(t_n+h\tau)) = f(t_n, y(t_n))+\tau h \left\{ \frac{\partial f}{\partial t} + f\frac{\partial f}{\partial y} \right\}_{t=t_n, y=y_n} +O(h^2)$$

- 積分を実行すると

$$y(t_{n+1}) = y(t_n) + hf(t_n, y_n) + \frac{1}{2}h^2 \left\{ \frac{\partial f}{\partial t} + f\frac{\partial f}{\partial y} \right\}_{t=t_n, y=y_n} + O(h^3)$$

中点法 (2 次 Runge-Kutta 法)

- 2 次公式

$$\begin{aligned} k_1 &= hf(t_n, y_n) \\ k_2 &= hf(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_1) \\ y_{n+1} &= y_n + k_2 \end{aligned}$$

- このとき

$$k_2 = h\left\{ f(t_n, y_n) + \frac{1}{2}h\frac{\partial f}{\partial t} + \frac{1}{2}k_1\frac{\partial f}{\partial y} + O(h^2) \right\}$$

- したがって

$$y_{n+1} = y_n + hf(t_n, y_n) + \frac{1}{2}h^2 \left\{ \frac{\partial f}{\partial t} + f\frac{\partial f}{\partial y} \right\}_{t=t_n, y=y_n} + O(h^3)$$

高次の Runge-Kutta 法

- 3 次 Runge-Kutta 法

$$\begin{aligned} k_1 &= hf(t_n, y_n) \\ k_2 &= hf(t_n + \tfrac{2}{3}h, y_n + \tfrac{2}{3}k_1) \\ k_3 &= hf(t_n + \tfrac{2}{3}h, y_n + \tfrac{2}{3}k_2) \\ y_{n+1} &= y_n + \tfrac{1}{4}k_1 + \tfrac{3}{8}k_2 + \tfrac{3}{8}k_3 \end{aligned}$$

- 4 次 Runge-Kutta 法

$$\begin{aligned} k_1 &= hf(t_n, y_n) \\ k_2 &= hf(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_1) \\ k_3 &= hf(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}k_2) \\ k_4 &= hf(t_n + h, y_n + k_3) \\ y_{n+1} &= y_n + \tfrac{1}{6}k_1 + \tfrac{1}{3}k_2 + \tfrac{1}{3}k_3 + \tfrac{1}{6}k_4 \end{aligned}$$

- 4 次までは次数と f の計算回数が等しい

計算コストと精度

- 実際の計算では $f(t,y)$ の計算にほとんどのコストがかかる
- 計算回数と計算精度の関係

	1 次 (Euler 法)	2 次 (中点法)	3 次	4 次
計算精度	$O(h)$	$O(h^2)$	$O(h^3)$	$O(h^4)$
計算回数	N	$2N$	$3N$	$4N$

- 高次の Runge-Kutta を使う方が効率的
- どれくらい小さな h が必要となるか、前もっては分からない
- 刻み幅を変えて $(h, h/2, h/4, \dots)$ 計算してみることが大事
  - 誤差の評価
  - 公式の間違いの発見